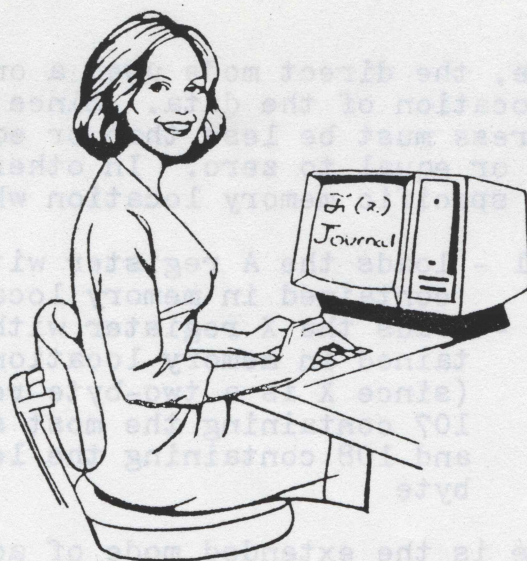


# F(x) Software Presents ...



The F(x) Journal

January 1983

This is the 4th issue of the F(x) Journal.

This month we have the 4th part of the tutorial on machine language, and useful peeks, pokes, and machine language ROM routines.

## MACHINE LANGUAGE TUTORIAL PART 4 - ADDRESSING MODES

The 6800 microprocessor has 6 different addressing modes, or ways of getting data into the CPU registers and accumulators. Each addressing mode has a particular name and method of operation.

The first mode to be discussed is the immediate mode. In this mode, the data is contained in the actual instruction; that is, the byte(s) following the instruction in memory are the actual data.

Example: LDAA #8 - loads the A register with the value of 8  
LDX #1257 - loads the X register with the value of 1257

Note that the "#" denotes immediate addressing.

The next mode, the direct mode uses a one byte address to specify the location of the data. Since only one byte is used, the address must be less than or equal to 255 and greater than or equal to zero. In other words, the byte points at a specific memory location which contains the needed data.

Example: LDAA 251 - loads the A register with the data contained in memory location 251

LDX 107 - loads the X register with the data contained in memory location 107 and 108 (since X is a two-byte register) with 107 containing the most significant byte and 108 containing the least significant byte

The next mode is the extended mode of addressing. The extended mode works much like the direct mode except a two byte address is used, meaning that any part of memory may be addressed.

Example: LDAA 16383 - loads the A register with the data contained in memory location 16383

LDX 12878 - loads the X register with the data contained in memory locations 12878 and 12879, with 12878 containing the most significant byte and 12879 containing the least significant byte.

The next mode is the indexed mode of addressing. In the indexed mode, the X register plus an offset is used to point at a specific memory location where the data is to be found. This mode may be used anywhere in memory since the X register is a two byte register.

Example: LDAA 4,X - loads the A register with the contents of the memory location pointed at by (X plus 4); if X contained 10121, then A would be loaded the contents of memory location 10125

LDX 0,X - loads the most significant byte of X with the data contained in memory location X, loads the least significant byte with the data contained in memory location X plus 1; if X contained 11245, then the most significant byte of X would be loaded with the contents of 11245 and the least significant byte of X would be loaded with the contents of 11246

Another mode of addressing, the relative mode, is used mainly by branch-type instructions, such as BNE (branch if not equal to zero) and BEQ (branch if equal to zero). In

Data Register B - 4 of the lines are used to complete the joystick/keypad matrix

b6 - controls GMO input of VDG

b7 - selects Alpha or Graphics

Control Register B - Control line 1 inputs field sync and

passes it to the MPU as an interrupt signal

Control line 2 generates the sound

oscillator

#### Special Memory Locations

41452 (\$A1EC) - Screen Save Flag. 0 - 512 to 1023, 255 - 0 to 511

41397 (\$A1B5) - COLOR byte

41398 (\$A1B6) - SHAPE type

41155 - 41388 (\$A0C3 - \$A1AC) - Variable table

40960, 40961 (\$A000, \$A001) - Cursor Pointer

41984, 41985 (\$A400, \$A401) - End of Basic Program Pointer

41009, 41010 (\$A031, \$A032) - Subscripted Variable Pointer

41446, 41447 (\$A1E6, \$A1E7) - End of Memory Pointer

40967, 40968 (\$A007, \$A008) - Lowest byte to R/W to/from Tape

40969, 40970 (\$A009, \$A00A) - Highest byte to R/W to/from Tape

508 (01FC) - I60 - if non-zero, causes an immediate JSR

to routine whose address is stored at I60J

453,454 (\$01C5, \$01C6) - I60J - JSR address if I60 is non-zero

509 (\$01FD) - ISEC - if non-zero, causes an immediate JSR

to routine whose address is stored at ISECJ

ISEC is tested once every second

455,456 (\$01C7, \$01C8) - ISECJ - JSR address if ISEC is non-zero

504 (\$01F8) - T60 - incremented every interrupt (every 1/60 sec.)

507 (\$01F8) - TIME - keeps count of 1/60 seconds

505 (\$01F9) - SECOND - incremented every second

506 (\$01FA) - MINUTE - incremented every 60 seconds

## ROM ROUTINES

Enable motor and audio.

SET-UP: none

CALL: CALL 34040, JSR \$84F8

Disable motor.

SET-UP: none

CALL: CALL 34061, JSR \$850D

CSAVE indicated memory.

SET-UP: 40967,40968 (\$A007, \$A008) - lowest byte to save

40969,40970 (\$A009, \$A00A) - highest byte to save

41446,41447 (\$A1E6, \$A1E7) - highest byte to save

CALL: CALL 34141, JSR \$855D

CLOAD indicated memory.

SET-UP: 40967,40968 (\$A007, \$A008) - lowest byte to load

40969,40970 (\$A009, \$A00A) - highest byte to load

41446,41447 (\$A1E6, \$A1E7) - highest byte to load

CALL: CALL 34228, JSR \$85B4

Clear screen to dark green.

SET-UP: none

CALL: CALL 17046, JSR \$4296

Enter machine language monitor.

SET-UP: none

CALL: CALL 28672, JSR

Move memory block.

LIMITS: Block to be moved is 255 bytes or less.

Destination address must be less than source address  
if the two overlap, otherwise data will be lost.

SET-UP: Acc B - number of bytes to move

41001,41002 (\$A029, \$A02A) - destination address

41003,41004 (\$A02B, \$A02C) - source address

CALL: JSR \$7700

Set screen to all one code.

SET-UP: Acc A - code to go to screen

CALL: JSR \$4298

Get input from joystick/keypads.

SET-UP: none

CALL: JSR \$41BE - left controller

JSR \$41D9 - right controller

RETURNS: Carry flag of status register,

if the carry is clear then no key pressed,

if the carry is set then the ASCII code for the key  
pressed is stored at \$01F2

Get input from keyboard.

SET-UP: none

CALL: JSR \$80CF

RETURNS: ASCII code for key pressed in Acc A, if Acc A  
equals 0 then no key was pressed.

Output to screen.

SET-UP: Acc A - code to output

(\$A000, \$A001) - screen address

CALL: JSR \$8473